Developing a voice-spelling alphabet for PDAs

by J. R. Lewis
P. M. Commarford

A persistent problem with personal digital assistants (PDAs) is the difficulty of entering data into the devices. The best current solutions to the problem are small soft keyboards and constrained handwriting recognizers. Another solution is use of speech. PDAs do not yet have the power to support full speech dictation, but they do have sufficient power to support voice spelling. Voice-spelling problems include the high acoustic confusability between certain letters of the alphabet and the difficulty of memorizing code words for the letters of the alphabet. This paper describes several User-Centered Design studies conducted to develop a voice-spelling alphabet for PDAs that overcomes these problems, including: (1) the development of a model of user performance to assess the potential of voice spelling as an alternate input method for PDAs, (2) Web-based surveys for determining the words that people tend to associate with the letters of the alphabet, (3) accuracy experiments used to tune the final voice-spelling alphabet, and (4) the development of a graphical user interface for displaying code words as a prompt when voice spelling is used. The results of these studies suggest that it would be worthwhile to develop a working voice-spelling system for PDAs in the future.

The primary goal of this research was to investigate the feasibility of voice spelling as an input method for personal digital assistants (PDAs). Voice spelling is far from the leading edge of current speech research and development at IBM, but we did not want to dismiss its potential usefulness out of hand. These studies are illustrative of some of the types of low-cost User-Centered Design studies that practitioners can conduct before committing the resources to build a working version of complex software. We have not yet had an opportunity to evaluate a working version of a PDA-based voice-spelling system, so it is possible to question the value of these preliminary studies. In response to such a question, we hold that conducting these types of studies acts as a filter to stop development of systems that have no chance to be competitive. Passing these filters does not assure success, but not passing them is a strong indicator of likely failure.

Note that even if research in the potential usefulness of voice spelling indicated likely competitiveness with the current input methods of stylized handwriting and soft-keyboard tapping, voice spelling would not be able to replace existing methods. Those persons who have achieved a high level of expertise in current methods would be unlikely to switch methods. Also, there are many social situations in which handwriting or soft-keyboard tapping would be acceptable but audible voice spelling would not.

In contrast, a common complaint about current input methods for PDAs is that they are quite slow. It is this complaint that motivated our explorations into the feasibility of voice spelling as an alternative (but not replacement) input method.

Current PDA input rates are slow. In the last decade both the functionality and popularity of PDAs have risen dramatically. These devices have many advantages, including small size, low weight, and extreme mobility. There are two primary methods of data input for PDAs—tapping a small virtual (soft) keyboard and using highly constrained handwriting recognizers such as Grafitti** or Unistrokes**. The current input speeds for these methods, however, are substantially slower than the typical standard keyboard input rates achieved with a personal computer.

Most PDA users have experience with personal computers and are familiar with the standard QWERTY keyboard, with which experts can type data at rates of approximately 55 words per minute (WPM)[1,2] with nearly perfect accuracy. As described below, research has shown that the input rates for various handwriting recognition systems and virtual keyboards are substantially slower.

Handwriting recognition. Paper-and-pencil hand-printing speeds range from approximately 12 to 23 WPM, and cursive handwriting speeds are typically in the 16 to 30 WPM[3] range, so these, necessarily, provide estimates of the upper limits for this sort of text entry. One experiment[4] using two discrete printing recognizers and a 9.5-inch tablet found a mean text entry speed of 17.1 WPM. Participants in that study received instructions to aim for both accuracy and speed but to ignore mistakes as they entered short phrases with no punctuation. The mean recognition accuracy was 92 percent when constraining the recognizer to lowercase letters and 90 percent for uppercase and lowercase letters.

MacKenzie and Zhang[5] found high (95.8 percent) recognition accuracy for experienced users of the Graffiti handwriting recognition system when writing the letters of the alphabet but did not report the entry speeds. MacKenzie et al.[6] found that users aiming for both speed and accuracy while ignoring mistakes could print 16.3 WPM (short phrases with no punctuation) onto a tablet using the Microsoft character recognizer and a tablet interface, but with an 8.1 percent error rate.

For participants using a PDA rather than a tablet to input addresses, simple URLs, and short notes (with and without punctuation), Sears and Arora[7] reported a much slower text entry rate for the Graffiti recognizer (WPM equal to 4.95 with 5 percent residual errors). They found that participants using the Jot** recognizer were able to produce 7.74 WPM with an 8 percent residual error rate. Participants in the experiment received instruction to balance speed and accuracy and consequently corrected some, but not all, errors.

Fleetwood et al.,[8] found that experts (at least three months ownership of PDAs running Palm OS**) using Grafitti were able to enter uncorrected text (a phone number and two short phrases without capitalization or punctuation) into a PDA at a rate of 20.69 WPM and that novices could enter the text at a rate of 6.82 WPM. They, however, found a high error rate (9 percent), with no significant differences in errors between the groups.

Each of the previously mentioned studies reported entry rates for uncorrected text or text in which participants made some but not all corrections. Kleid and Bonto[9] asked users to enter a rather complex set of letters, numbers, and special characters (a person's contact information) using Graffiti on a 6-inch (diagonal) screen. Participants were to attempt 100 percent accuracy and to use any editing tools that they felt would be helpful. Under these conditions, participants only entered 1.98 corrected words per minute (CWPM).

Typing with soft keyboards. Research has shown stylus tapping on a virtual QWERTY keyboard to be slightly faster than handwriting recognition, but these rates are still slow in comparison to standard typing speeds. Kleid and Bonto[9] had participants use a soft keyboard to enter the previously described set of complex text with 100 percent accuracy and observed mean throughput rates of only 5.17 CWPM.

Zha and Sears[10] reported that participants could input text typical of an e-mail note on a PDA at a rate of 12.62 WPM with an average residual accuracy of 96 percent using a soft keyboard. Participants in the study were to balance speed and accuracy and typically corrected some, but not all, errors.

Fleetwood et al.[8] measured the entry rate of uncorrected text into a PDA as 17.91 WPM for experts and 15.38 for novices, with a 2 percent error rate across groups. The faster input speed observed in this study was most likely a result of the input stimuli. These participants entered a set of three short phrases with no punctuation or capitalization and no requirement to switch between letters and digits.

Using a tablet, MacKenzie et al.[6] found that users could type text (short phrases with no capitalization or punctuation) at a rate of 22.9 WPM with 99 percent accuracy. Participants tapping on a full-sized paper QWERTY layout had text entry rates of 20.2 WPM.[11] Participants received instructions to ignore mistakes.

Voice throughput may offer a faster throughput rate for handheld devices than currently available options.

What is the target throughput? The aforementioned research suggests faster rates of PDA throughput for virtual keyboards than for handwriting recognizers. The study conducted by Zha and Sears in 2001 provides the most realistic estimate of PDA throughput with a soft keyboard. These researchers asked participants to use a PDA to input a 45-word passage characteristic of a short business e-mail message. They found a mean input rate for this task of 12.62 WPM, with a 4 percent error rate. By accounting for errors and assuming a fairly rapid correction speed, it seems reasonable to set the benchmark for the true throughput rate for soft keyboard input at 12 CWPM.

The value of 12 CWPM seems appropriate for novice users of soft keyboards, but might not be correct for experts. There is no existing study of expert use of soft PDA keyboards with error correction during the performance of realistic tasks, but MacKenzie and Zhang[12] found a 40 percent improvement in the throughput of a soft tablet keyboard over an extended period of use. If this improvement is applied to the novice rate of 12 CWPM, it seems reasonable to set the estimated value for expert usage at 16.8 CWPM. Further, the input rate reported by Fleetwood et al.[8] (17.9 WPM) would probably be attenuated to some degree had participants entered more representative text and corrected all errors, thus providing additional support for 16.8 CWPM as a reasonable estimate of expert performance.

The potential of voice input. In recent years, a new method of entering data into a personal computer has become available—speech dictation. Voice throughput is not generally as fast as typing with a full-sized QWERTY keyboard; however, it may offer a faster throughput rate for handheld devices than the currently available options. Lewis[13] defined true throughput as the number of correct words produced per minute. In a study of two commercially available speech dictation products, he found that participants could achieve rates of 31.0 CWPM with multimodal (manual and vocal) correction and 19.0 CWPM with voice-only correction.

In the Lewis study,[13] the average speaking rates during dictation of prepared materials were just over 100 WPM, with multimodal correction speeds of 13.2 seconds per correction. Note, however, that the correction speeds Lewis measured were full-word corrections, resulting from system misrecognitions of user dictation. Correcting the misrecognition of a single character would presumably be much quicker, requiring a minimum of two acts (tapping the Backspace key, followed by tapping the appropriate key or saying the appropriate code; saying "backspace," then tapping or saying the appropriate key or code). If a user made a correction inside of existing text, an additional stylus tap would be required to place the insertion point. At 12 WPM, participants have demonstrated the ability to tap one key per second on a soft keyboard. Multimodal (stylus plus voice) correction could conceivably be even faster. Therefore, corrections could be as fast as two seconds per correction (a probably unachievable best case) and, in the worst case, would probably be no slower than 20 seconds per correction. Because it would take some time for other processes (detecting the error, insertion point placement if necessary, etc.), five seconds per correction seems to be a reasonable estimate of the multimodal correction rate for voice spelling with a PDA.

Users would benefit greatly if system designers could embed voice recognition technology into the PDA environment in a way that allowed throughput rates similar to those observed by Lewis.[13] Resource limitations of these handheld devices, however, will probably prevent the high levels of recognition accuracy reached by desktop software; in fact, these limitations will probably prevent the introduction of speech dictation for the immediate future. In addition, multimodal correction on a PDA will include the use of less efficient methods of input (soft keyboard or handwriting recognizer) than those used with desktop systems.

It is reasonable to consider the efficiency of another method of entering data by voice in a PDA environment: voice spelling. Because recognition of spoken letters is notoriously poor, users would say codes for each letter of the alphabet. For example, rather than saying "d," the user would say "dog." It would be possible to present a reminder display in voice-spelling mode for users who have not memorized the codes. Further, this display could accept user entries by tapping the codes, thus allowing multimodal input and correction.

Users would certainly be able to produce more uncorrected words per minute via dictation than by voice spelling. However, because of the limited grammar set, voice spelling would probably achieve higher levels of system recognition accuracy than dictation, reducing the need for correction. The prior modeling in the Lewis study suggests that system recognition accuracy is a more important determinant of true throughput than speaking rate. By modeling true throughput (CWPM) for voice spelling based on system accuracy, time per correction, and speaking rate, we can compare voice spelling to reported throughput rates for other PDA input methods.

Phase 1: A model of voice spelling performance

Before committing significant resources to the development of a voice-spelling method for PDAs, we decided to model the expected true throughput for voice spelling based on the available data. This section of the paper describes performance modeling conducted to:

- Compare expert spelling throughput for 100- and 150-WPM speakers
- Compare novice spelling throughput for 100- and 150-WPM speakers
- Determine the points on these performance curves that first match or beat the soft keyboard benchmark of 12 CWPM (for novices) and 16.8 CWPM (for experts)

The results of this first investigation were critical because they would determine whether it would be reasonable to continue development of a voice-spelling method.

Method. We created four models of true throughput for voice spelling by crossing the independent variables of user (novice and expert) and speaking rate (100 and 150 WPM). Each model contained a range of system recognition accuracies from 90 percent to 100 percent and a range of correction times from 2 to 20 seconds. The levels of speaking rate, correction speed, and recognition accuracy were consistent with the range of values expected from earlier studies of desktop dictation and the known differences between voice spelling and dictation. The models assumed that the recognized letters would appear instantaneously on the PDA screen as the user spoke. Although current technology would not allow this real-time processing for the large vocabulary necessary for PDA dictation, a grammar for a voice-spelling alphabet would be sufficiently small for the system to display results with no detectable latency.

The expert voice-spelling model assumed complete automaticity in the assignment of the letters to their respective codes (in other words, expert users would need no processing time to match letters to their codes or to retrieve them from short-term memory). The novice models assumed a 230-ms eye movement time to locate each letter in the passage on a spelling vocabulary reminder display. The 230-ms eye movement time is consistent with that given as the typical or "middleman" time by Card et al.[14]

Expert speller model. The expert speller model allowed estimation of the voice-spelling throughput rates for a speaker for whom the spelling letter codes have become automatic. Table 1 shows the expected voice-spelling throughput for expert spellers, speaking 150 WPM and 100 WPM at varying levels of recognition accuracy and varying correction speeds. The *bold* numbers in the table indicate the point at which voice spelling becomes competitive with novice soft keyboard input (in other words, exceeds 12 CWPM). The ***bold italic*** numbers indicate the point at which voice spelling becomes competitive with the estimated expert input speed of 16.8 CWPM.

Table 1 shows that at five seconds per correction, system recognition accuracy would need to be about 90 percent for voice spelling to be a competitive alternative to soft keyboard input (novice speed) for fast-speaking expert spellers, and would need to be about 96 percent to beat the expert stylus-tapping speed. With 97 percent system accuracy, five seconds per correction, and fairly fast speech, voice spelling could be more than 1.5 times as productive as using

Table 1 Model of expected throughput (CWPM) rates for expert spellers

Speaking Rate	Correction Speed	Recognition Accuracy in Percent										
		100	99	98	97	96	95	94	93	92	91	90
150 WPM	**2 sec**	27.16	25.86	24.69	23.62	22.63	21.73	20.89	20.12	19.40	18.73	***18.10***
	5 sec	27.16	24.14	21.73	19.75	***18.10***	16.71	15.52	14.48	13.58	12.78	**12.07**
	10 sec	27.16	21.73	***18.10***	15.52	13.58	**12.07**	10.86	9.88	9.05	8.36	7.76
	15 sec	27.16	***19.75***	15.52	**12.78**	10.86	9.45	8.36	7.49	6.79	6.21	5.72
	20 sec	27.16	***18.10***	**13.58**	10.86	9.05	7.76	6.79	6.03	5.43	4.94	4.53
100 WPM	**2 sec**	18.10	17.52	***16.97***	16.46	15.97	15.52	15.09	14.68	14.29	13.93	**13.58**
	5 sec	***18.10***	16.71	15.52	14.48	13.58	12.78	**12.07**	11.43	10.86	10.35	9.88
	10 sec	***18.10***	15.52	13.58	**12.07**	10.86	9.88	9.05	8.36	7.76	7.24	6.79
	15 sec	***18.10***	14.48	**12.07**	10.35	9.05	8.05	7.24	6.58	6.03	5.57	5.17
	20 sec	***18.10***	**13.58**	10.86	9.05	7.76	6.79	6.03	5.43	4.94	4.53	4.18

Table 2 Model of expected throughput (CWPM) rates for novice spellers

Speaking Rate	Correction Speed	Recognition Accuracy in Percent										
		100	99	98	97	96	95	94	93	92	91	90
150 WPM	**2 sec**	18.47	17.87	17.30	16.76	16.26	15.79	15.34	14.92	14.52	14.14	**13.79**
	5 sec	18.47	17.03	15.79	14.72	13.79	12.96	**12.23**	11.58	11.00	10.47	9.98
	10 sec	18.47	15.79	13.79	**12.23**	11.00	9.98	9.14	8.43	7.83	7.30	6.84
	15 sec	18.47	14.72	**12.23**	10.47	9.14	8.12	7.30	6.63	6.08	5.61	5.20
	20 sec	18.47	**13.79**	11.00	9.14	7.83	6.84	6.08	5.46	4.96	4.55	4.20
100 WPM	**2 sec**	13.79	13.44	13.12	12.81	12.52	**12.23**	11.96	11.71	11.46	11.22	11.00
	5 sec	13.79	12.96	**12.23**	11.58	11.00	10.47	9.98	9.55	9.14	8.77	8.43
	10 sec	13.79	**12.23**	11.00	9.98	9.14	8.43	7.83	7.30	6.84	6.44	6.08
	15 sec	**13.79**	11.58	9.98	8.77	7.83	7.06	6.44	5.91	5.46	5.08	4.75
	20 sec	**13.79**	11.00	9.14	7.83	6.84	6.08	5.46	4.96	4.55	4.20	3.90

a soft keyboard at the novice level of speed, and about 1.2 times as productive as the estimated expert use of a soft keyboard.

For slower speakers, recognition accuracy would have to be at least 94 percent to compete with novice stylus tapping and would have to be about 99 percent to compete with expert stylus tapping. At 20 seconds per correction, recognition accuracy would have to be nearly perfect for voice spelling to be a competitive alternative.

Novice speller model. The novice speller model allows the estimation of throughput rates for speakers who are just learning to voice spell and have not yet memorized the letter codes. Table 2 shows the expected throughputs for novice spellers at 150 WPM and at 100 WPM with varying levels of system recognition accuracy and varying correction speeds. This model assumes that the user will need to visually scan a small display for each letter code before speaking the code. We further assume that the user does not need a visual reminder for *period* or *space*. The bold numbers in the table indicate the point at which voice spelling becomes competitive with soft keyboard input.

The expected novice speller data show that if recognition accuracy is 94 percent or greater and corrections take five seconds, voice spelling would immediately be as efficient as tapping on a virtual keyboard for fast speakers. Recognition accuracy would need to be very high (98 percent or greater) for this new method to instantly benefit slower speakers. With 98 percent system accuracy, fast-speaking novice spellers would see a benefit of nearly 33 percent with voice spelling (15.79 CWPM for voice spelling vs. 12 CWPM for soft keyboard entry).

Voice-spelling modeling conclusions. The results of this modeling effort show that as long as the recognition accuracy of a voice-spelling system was 96 percent or greater, voice spelling could be as efficient as any other input method currently available for modern PDAs. The model also suggests the importance of minimizing learning time and processing effort, thus allowing users to obtain expert status after minimal exposure to the codes. In addition, consideration of the components of the models indicates that allowing novice users to scan a visual display for quick access to letter codes is a critical aspect of any voice-spelling input method. Without a display for reference, it is clear that novice throughput speeds would be unacceptably slow.

Phase 2: Web-based survey to collect potential code words

Having determined the feasibility of voice spelling as an input method for PDAs in Phase 1, we had to cope with the constraints of presenting code words on a small display. Where possible, we wanted the displayed code word to be from the military alphabet (more officially known as the International Civil Aviation Organization code—the code that starts with *alpha* and *bravo* and ends with *Yankee* and *Zulu*). This design decision was based on the following considerations:

- These codes have a development history that includes acoustic distinctiveness.
- Although few people in the general population have memorized these code words, there is no other set of reasonably well-known code words with similar properties.
- There are some special populations (military, pilots, police, emergency workers, ham radio operators, etc.) whose members have memorized the code words.
- These code words have no disadvantage compared to others because users who do not know them will read them from a display.

Unfortunately, over 25 percent of the code words in the military alphabet have a length that makes them unsuitable for presentation in a matrix of code words displayed on a PDA screen. If a proportional font is used, words with five or fewer letters are generally acceptable, as are words with six letters when one of the letters is *i* (the narrowest letter). Specifically, the words of questionable length were *Charlie*, *foxtrot*, *November*, *Quebec*, *uniform*, *whiskey*, and *Yankee*.

The purpose of the research in this phase was to find out what words come first to a person's mind when asked to produce a word for each letter of the alphabet. The goal was to find alternatives for the long words listed above which would be short enough to display on a PDA screen. In one experiment participants provided words without constraint. In a separate experiment, participants provided first names for each letter. The reason for constraining participants in this way was to see if constraining responses to names led to more consistent responses than unconstrained production. If so, then the constrained set would be more useful as a source for alternative code words. If not, then the unconstrained set would be more useful.

Method. This subsection describes the participants, materials, and procedure used in this study.

Participants. The source for participants in each experiment was a set of 400 IBM employees (800 employees in all, 400 for each experiment), selected at random from an internal e-mail directory of all of the IBM employees in the United States and sent an e-mail invitation to participate. Of the employees invited to participate, 103 (26 percent) responded to the invitation for the first experiment (no constraints), and 120 (30 percent) responded to the invitation for the second experiment (names for letters).

Materials. We used the version of WebSurveyor** from the IBM User-Centered Design toolset to construct a Web-based form for the evaluation. The form simply provided a space by each letter of the alphabet (arranged vertically on the page in alphabetical order), with instructions appropriate for the specific experiment (either to provide any word for each letter or to provide only first names).

Procedure. After receiving the e-mailed invitation, participants clicked a link in the message that brought up the WebSurveyor page containing the survey that the participant was to take. The participants read an introduction explaining the purpose of the survey, then completed and submitted the form.

Results. We conducted a number of comparisons to assess the distributions of potential code words for the unconstrained and constrained-to-names conditions.[15] The data suggested that the unconstrained word distribution was preferable to the names distribution because, contrary to our expectation, the overall consistency of unconstrained responses was not poorer than the responses in the constrained-to-names distribution. Furthermore, the occurrence of cases in which participants were not able to provide a response for the names distribution was sig-

Table 3 Top three word code candidates for each letter of the alphabet (members of the military alphabet marked with an asterisk)

Word	Percent	Cumulative Percentage	Word	Percent	Cumulative Percentage
Apple	67.0	67.0	Nancy	49.5	49.5
Alpha*	11.7	78.6	no	5.8	55.3
Able	4.9	83.5	nice	5.8	61.2
Boy	40.4	40.4	open	21.4	21.4
Baker	9.6	50.0	Oscar*	18.4	39.8
Bravo*	9.6	59.6	orange	12.6	52.4
Cat	38.5	38.5	Paul	23.5	23.5
Charlie*	37.5	76.0	peter	13.7	37.3
Charles	2.9	78.8	Papa*	5.9	43.1
Dog	66.0	66.0	queen	31.1	31.1
David	13.6	79.6	quick	11.7	42.7
Delta*	8.7	88.3	Quebec*	8.7	51.5
Elephant	21.7	21.7	Robert	17.5	17.5
Edward	14.2	35.8	rabbit	7.8	25.2
Echo*	12.3	48.1	Romeo*	6.8	32.0
Frank	34.0	34.0	Sam	37.5	37.5
Fox	21.4	55.3	snake	3.8	41.3
Foxtrot*	4.9	60.2	star	3.8	45.2
George	25.0	25.0	Tom	35.3	35.3
Girl	16.3	41.3	tango*	6.9	42.2
Good	9.6	51.0	Thomas	4.9	47.1
Help	13.6	13.6	under	20.0	20.0
Henry	13.6	27.2	uncle	14.7	34.7
Harry	11.7	38.8	ugly	8.0	42.7
Igloo	13.5	13.5	vertical	49.0	49.0
India*	12.5	26.0	victory	16.7	65.7
Indian	7.7	33.7	violin	4.9	70.6
Jack	16.5	16.5	water	9.0	9.0
Jump	11.7	28.2	William	9.0	18.0
John	9.7	37.9	whiskey*	8.0	26.0
King	19.6	19.6	x-ray*	64.6	64.6
Kite	16.7	36.3	xylophone	16.7	81.3
Kilo*	10.8	47.1	Xerox	11.5	92.7
Larry	22.3	22.3	yellow	45.5	45.5
Love	10.7	33.0	yes	13.1	58.6
Lima*	6.8	39.8	Yankee*	10.1	68.7
Mary	42.7	42.7	zebra	75.2	75.2
Mike*	8.7	51.5	zoo	5.9	81.2
Man	6.8	58.3	zero	5.0	86.1

nificantly greater than for the unconstrained distribution (in which the problem never occurred).

Table 3 shows the top three code-word candidates for each letter of the alphabet from the unconstrained distribution. Also included in the table are the percentage of production for each word and the cumulative percentage for the top three candidates for each letter. Words marked with an asterisk are members of the military alphabet. In 18 cases (69 percent), the code word from the military alphabet was one of the top three candidates.

Using the data from Table 3, we selected the following replacements for the longer code words from the military alphabet (with the selection percentage for the new word appearing in parentheses): *cat* (38.5 percent) for *Charlie*; *frank* (34.0 percent) for *foxtrot*; *north* (1.9 percent) for *November*; *queen* (33.0 percent) for *Quebec*; *under* (20.0 percent) for *uniform*; *water* (9.0 percent) for *whiskey*; *yoyo* (5.1 percent) for *Yankee*.

For two letters (*N* and *Y*), none of the top three alternatives from Table 3 were suitable because of a combination of length and/or acoustic distinctiveness. Our goal for acoustic distinctiveness was to attempt to create a set of words that differed in their usage of the major vowel sounds. The top choice for *Y* (*yellow*) was too long to fit in the planned matrix. The top choice for *N* (*Nancy*) used an initial vowel sound that already had representation in the set (*tango*). The second choices (*no* and *yes*) were unsuitable because of the likelihood of their being active as commands during voice spelling. The vowel for the third choice for *N* (*nice*) was already present in the set (*Mike*). Therefore, we selected alternative words of high acoustic distinctiveness—*north* and *yoyo*.

Although *Juliet* was not too long, we had a concern that its acoustic similarity to *period* might cause recognition problems, prompting us to seek an alternative to include in the initial test set (along with *Juliet*). The vowels for the *J* words in Table 3 all had prior representation in the code set, so we selected *Jane* to represent *J* because it was the only word in the set of letter codes with a long *a* vowel.

We made the trade-off to use code words that were not one of the top three candidates as long as they were acoustically distinct because users of the target system would have access to a display of the code words, making it a better design decision to maximize accuracy rather than immediate recall. The words selected for acoustic distinctiveness that were not in the top three choices were all in the top ten (*yoyo* was fifth, *north* was seventh, *Jane* was eighth).

Phase 3: Accuracy experiments

In this phase we conducted four experiments to develop and tune speech recognition grammars for voice spelling on devices with small displays. There were two compelling reasons to conduct these accuracy experiments, even though we had based the majority of our spelling code words on the military alphabet.

First, there were enough new words in the modified voice-spelling alphabet to justify studies of its recognition accuracy. Second, a comprehensive voice-spelling system requires the recognition of digits, punctuation commands, and commands that control the location of the insertion point. It is possible that words selected for the modified voice-spelling alphabet might be acoustically confusable with these commands.

General method. The participants were four males and four females (all adult native speakers of American English) who provided recordings of the test items (8-kHz sampling rate, 16-bit dynamic range). We used the IBM WebSphere* Voice Server SDK Version 2.0 to decode the recorded scripts. The Voice Server does not run on a PDA, but because it is designed for recognition of telephony audio, it uses audio recorded with a sampling rate of 8-kHz—the same as standard PDA audio. This is considerably lower than the 22-kHz sampling rate used for speech recognition in standard desktop systems.

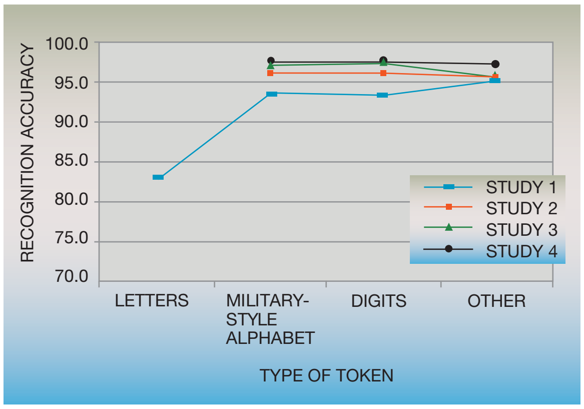
The recorded test script included the letters of the alphabet, the custom voice-spelling alphabet, digits, punctuation, and commands for controlling the location of the insertion point. The test grammar (the set of all items available to be recognized by the system) included entries for all of these items, and also included the long items from the standard military alphabet so that users who had memorized the military alphabet could use it (see Appendix A for this initial grammar). The primary dependent measure in these experiments was the simple command accuracy for each item in the test script, organized by item type.

Users provided recordings of all test phrases (letters, code words, and other phrases) one phrase at a time, recorded with a close-field noise-canceling headset microphone (Andrea Electronics NC-61) in a quiet office environment at a sampling rate of 8 kHz. The recordings were stored in a standard desktop computer as audio files.

Experiment 1. The full grammar in this experiment involved all the letters of the alphabet.

Motivation. The purpose of the first experiment was to investigate the accuracy of a voice-spelling gram-

Figure 1 Summary of results of recognition accuracy experiments



mar that included the letters of the alphabet, the first version of a customized voice-spelling alphabet, digits, punctuation, and commands for controlling the location of the insertion point.

Results and discussion. The accuracy of the voice-spelling alphabet was significantly (10.1 points) higher than that of the letters of the alphabet ($t(7) = 3.86$, $p = 0.01$). The accuracy of the other items (digits and punctuation/commands) was comparable to the accuracy of the military-style alphabet. (For a summary of the accuracy results of all four accuracy experiments, see Figure 1.)

The recognition accuracy for an item is the likelihood of the system correctly recognizing the item when a user speaks it. The conditional probability is the likelihood that the item produced by the recognizer is the one that the user actually spoke. Sometimes the accuracy and conditional probability for an item can be very different. For example, in Experiment 1 the accuracy for the letter *F* was only 25 percent because the system frequently misrecognized it as *S*. Its conditional probability, however, was 100 percent because whenever the system returned an *F*, the spoken letter was *F*.

The conditional probabilities for items that the system did not recognize correctly can be especially useful for designing intelligent correction schemes for voice-spelling systems.[16] In all cases but one, the conditional probability for the items in the new military-style voice-spelling alphabet was 100 percent. The exception was the word *Jane*, which the system produced in error once for *A* and once for *G*.

There were 17 problem test items, 10 of which were letters. For most of the seven remaining problem items, the majority of misrecognitions involved letters. This indicated that overall recognition accuracy might improve as a result of removing letters from the grammar (see Experiment 2). The problem items never misrecognized as letters were *papa*, *period*, and *dot*. *Papa* and *dot* had fairly high rates of having nothing returned from the recognizer. *Period* had fairly high rates of low confidence responses.[17]

Experiment 2. The effect of removing the letters of the alphabet was the focus of this experiment.

Motivation. The results of the first experiment provided a baseline for evaluating the relative accuracy of a grammar that did not include the letters of the alphabet. The purpose of Experiment 2 was to investigate the accuracy of a grammar that was the same as that used in Experiment 1 except for the absence of the letters of the alphabet. Recognition accuracy of the letters of the alphabet is notoriously poor (as seen in Experiment 1), but unless removing these letters from the grammar resulted in improved recognition for other items, then the best strategy would be to leave them available for use. In contrast, if removing them led to improved recognition for other items, then the best strategy would be their removal.

Results and discussion. The overall accuracy in Experiment 2 was somewhat better than the comparable accuracy in Experiment 1 ($t(7) = 2.22$, $p = 0.06$). The improvement in the accuracy of the items in the military-style alphabet achieved marginal statistical significance ($t(7) = 2.02$, $p = 0.08$). The most frequently occurring recognition problems in Experiment 2 were for the items *period* (37.5 percent accuracy), *dot* (62.5 percent), *papa* (75 percent), *five* (75 percent), and *plus* (75 percent). The accuracy of all other test phrases was 87.5 percent or higher.

It appeared that acoustic similarity between *period* and *Juliet* lowered the recognition accuracy for *period* because of substitutions of *Juliet*. The words *dot* and *papa* continued to have a tendency to fail to appear. There was some slight confusion between *five* and *seven* and *plus* and *slash*.

Experiment 3. This experiment is Phase 1 of tuning the grammar.

Motivation. Removing the letters of the alphabet from the grammar appeared to improve recognition accuracy, but some items continued to have relatively low accuracy. The purpose of Experiment 3 was to retest the grammar after tuning it (attempting to correct the remaining problems). There were no viable alternatives for *five* or *plus*, so the grammar remained the same for these items. The specific interventions for *Juliet*, *dot*, and *papa* were:

- Removed *Juliet* from grammar because of its acoustic likelihood to be confused with *period*.
- Provided additional support for recognition of *dot* in the context of Web URL and e-mail addresses. Specifically, the phrases added to the grammar for this purpose were *dot com*, *dot org*, *dot net*, *dot cat*, *dot Oscar*, and *dot north*.
- Replaced *papa* with *Paul* (the top alternative for the letter *P* from Phase 2) as the code word to display for *P* and added *Paul* to the grammar as a synonym for *papa*, which was still in the grammar.

Results and discussion. The overall accuracy for Experiment 3 improved by one point relative to Experiment 2, but this change was not statistically significant ($t(7) = 1.71$, $p = 0.13$). Compared to the results of Experiment 1, the difference was statistically significant ($t(7) = 2.94$, $p = 0.02$). The change to the grammar fixed the *papa* problem, but *period* and, to a lesser extent, *dot* continued to have problems. *Delta* produced the always active command *help* in two instances, indicating that the voice-spelling alphabet should include an alternate word for *D* (but there was no need to remove *delta* from the overall grammar).

The removal of *Juliet* from the grammar created some potential for confusion by expert spellers who had prior experience with the military alphabet. A user who said *Juliet* for *J* might be surprised at the result, but would have the display of code words available to discover the alternate code word. Leaving *Juliet* in the grammar would have had the more adverse consequence of interfering with the recognition of a frequently used punctuation mark (*period*), with no recourse for the user because there is no alternative code word available for *period*. Thus, the decision to remove *Juliet* from the grammar seemed to be the best way to address the design trade-off.

Experiment 4. This experiment is Phase 2 of tuning the grammar.

Motivation. In the final tuning step, we added custom pronunciations for the words that had continuing recognition problems (*period* and *dot*). We also replaced *delta* with *dog* in the custom voice-spelling alphabet and added *dog* to the grammar set to address the acoustic potential for confusion of *delta* and *help*.

Results and discussion. The overall accuracy for Experiment 4 improved by 0.7 percent relative to Experiment 3, but this change was not statistically significant ($t(7) = 1.0$, $p = 0.35$). Compared to the results of Experiment 1, the difference in overall accuracy was statistically significant ($t(7) = 2.97$, $p = 0.02$). Compared to Experiment 1, the increase in accuracy for the custom voice-spelling alphabet was statistically significant ($t(7) = 2.35$, $p = 0.05$). The changes to the grammar fixed the *period* and *delta* problems, and reduced, to some extent, the *dot* problem. See Appendix B for the final version of the grammar.

Summary of recognition results. Figure 1 shows the pattern of recognition accuracy results across the four experiments. For the final grammar, the overall recognition accuracy was 97.5 percent, with 97.6 percent for the custom voice-spelling alphabet, 97.5 percent for digits, and 97.4 percent for punctuation and insertion-point location commands.

Phase 4: Final code word matrix for a PDA voice-spelling interface

On the basis of the experiments conducted in Phase 3, the final grammar did not include the names of the letters of the alphabet, but included all code words from the custom voice-spelling and military alphabets (except for *Juliet*), punctuation, and cursor movement commands. The code words used in the final voice-spelling alphabet were short enough to fit on a small display; therefore, users would not have to memorize them. These words had very high recognition accuracies and high conditional probabilities, giving them the potential for a very effective voice-spelling system. Figure 2 illustrates one way to arrange the new custom voice-spelling alphabet for presentation on a small-screen device.

Conclusions

The studies described in this paper illustrate several types of User-Centered Design activities. The focus of Phase 1 was a set of user models for voice spelling, based on published human performance data.

Figure 2 Final code word matrix

Alpha	Bravo	Cat	Dog	Echo	Frank
Golf	Hotel	India	Jane	Kilo	Lima
Mike	North	Oscar	Paul	Queen	Romeo
Sierra	Tango	Under	Victor	Water	X-ray
Yoyo	Zulu	Space		"End Spell"	

Phase 2 employed an IBM User-Centered Design tool, WebSurveyor, to gather data on users' associations of words with the letters of the alphabet. Phase 3 was a series of designed experiments conducted to tune and finalize both the words to present to users in the graphical user interface of a PDA and the underlying recognition grammar for a comprehensive voice-spelling system. Phase 4 was the development of a specific user interface to display in such a voice-spelling system.

The modeling conducted in Phase 1 indicated that the accuracy of a voice-spelling system had to exceed 96 percent for fast-talking expert voice spellers to be competitive with expert key tapping (16.8 CWPM) on a soft keyboard (assuming five seconds per correction). For slower-speaking experts, the accuracy criterion was 99 percent. The minimum criterion was 94 percent accuracy for faster-speaking novices to surpass the expected novice tapping rate of 12 CWPM (assuming five seconds per correction). For slower-speaking novices, the criterion was 98 percent.

The overall accuracy from the final experiment in Phase 3 was 97.5 percent—exceeding the above criteria for fast speakers, but not for slow speakers. In practice the voice-spelling accuracy could be somewhat lower. During Phase 3, users provided the acoustic data through discrete speaking methods, recorded with a close-field noise-canceling headset microphone in a quiet office environment. In actual use, however, it is likely that users will speak continuously, running one letter into the next, increasing the likelihood of misrecognition errors. It is possible to use close-field noise-canceling microphones with most PDAs, but most built-in microphones are of lower quality, intended for use as far-field microphones with little or no ability to cancel noise. This situation suggests that successfully entering data into a PDA by voice spelling is likely to require the use of an auxiliary headset microphone, at least for the near future. Fortunately, the cost of these microphones has fallen considerably over the past few years. The potential for increased PDA data entry speeds (about 19.75 CWPM, assuming a fast-speaking expert with five seconds per correction compared to 16.8 CWPM for expert entry with a soft keyboard) could justify the purchase of such a microphone for some users, or provide an incentive for PDA manufacturers to include this type of microphone with their products.

The information gained from this series of studies suggests that voice spelling could, in some cases, be a viable alternative to current PDA input methods. As with current PDA input methods, certain circumstances would make voice spelling unacceptable. For example, high-noise environments (an airport), inappropriate social settings (a library), and certain individual differences (a foreign-language accent or speech dysfluencies) would prevent the success of such an input method. However, there are other circumstances in which voice spelling would be a desirable alterative. This is especially true for users who are comfortable speaking at a fairly rapid rate and for users with motor disabilities that make tapping or handwriting recognition difficult or impossible. Further, voice spelling would be possible when walking or when one hand is being used for another activity, whereas using other input methods would be difficult or impossible in these situations.

Appendix A: The initial voice-spelling grammar

```
#JSGF V1.0;
grammar voicespell;
//Copyright (c) 2002 IBM Corp. All Rights Reserved.
⟨a⟩ = a|alpha;
⟨b⟩ = b|bravo;
⟨c⟩ = c|charlie|cat;
⟨d⟩ = d|delta;
⟨e⟩ = e|echo;
⟨f⟩ = f|foxtrot|frank;
⟨g⟩ = g|golf;
⟨h⟩ = h|hotel;
⟨i⟩ = i|india;
⟨j⟩ = j|juliet|jane;
⟨k⟩ = k|kilo;
⟨l⟩ = l|lima;
⟨m⟩ = m|mike;
⟨n⟩ = n|november|north;
⟨o⟩ = o|oscar;
⟨p⟩ = p|papa;
⟨q⟩ = q|quebec|queen;
⟨r⟩ = r|romeo;
⟨s⟩ = s|sierra;
⟨t⟩ = t|tango;
⟨u⟩ = u|uniform|under;
```

```
⟨v⟩ = v|victor;
⟨w⟩ = w|water|whiskey;
⟨x⟩ = x|x ray;
⟨y⟩ = y|yankee|yoyo;
⟨z⟩ = z|zulu;
⟨upper⟩ = capital|uppercase;
⟨capslock⟩ = caps lock|capital|uppercase|lowercase;
⟨lower⟩ = lowercase;
⟨capping⟩ = ⟨upper⟩
  |⟨lower⟩
  |⟨capslock⟩
  ;
⟨lowerletter⟩ = [⟨lower⟩]⟨a⟩
  |[⟨lower⟩]⟨b⟩
  |[⟨lower⟩]⟨c⟩
  |[⟨lower⟩]⟨d⟩
  |[⟨lower⟩]⟨e⟩
  |[⟨lower⟩]⟨f⟩
  |[⟨lower⟩]⟨g⟩
  |[⟨lower⟩]⟨h⟩
  |[⟨lower⟩]⟨i⟩
  |[⟨lower⟩]⟨j⟩
  |[⟨lower⟩]⟨k⟩
  |[⟨lower⟩]⟨l⟩
  |[⟨lower⟩]⟨m⟩
  |[⟨lower⟩]⟨n⟩
  |[⟨lower⟩]⟨o⟩
  |[⟨lower⟩]⟨p⟩
  |[⟨lower⟩]⟨q⟩
  |[⟨lower⟩]⟨r⟩
  |[⟨lower⟩]⟨s⟩
  |[⟨lower⟩]⟨t⟩
  |[⟨lower⟩]⟨u⟩
  |[⟨lower⟩]⟨v⟩
  |[⟨lower⟩]⟨w⟩
  |[⟨lower⟩]⟨x⟩
  |[⟨lower⟩]⟨y⟩
  |[⟨lower⟩]⟨z⟩
  ;
⟨upperletter⟩ = ⟩upper⟩⟨a⟩
  |⟨upper⟩⟨b⟩
  |⟨upper⟩⟨c⟩
  |⟨upper⟩⟨d⟩
  |⟨upper⟩⟨e⟩
  |⟨upper⟩⟨f⟩
  |⟨upper⟩⟨g⟩
  |⟨upper⟩⟨h⟩
  |⟨upper⟩⟨i⟩
  |⟨upper⟩⟨j⟩
  |⟨upper⟩⟨k⟩
  |⟨upper⟩⟨l⟩
  |⟨upper⟩⟨m⟩
  |⟨upper⟩⟨n⟩
  |⟨upper⟩⟨o⟩
  |⟨upper⟩⟨p⟩
  |⟨upper⟩⟨q⟩
  |⟨upper⟩⟨r⟩
  |⟨upper⟩⟨s⟩
  |⟨upper⟩⟨t⟩
  |⟨upper⟩⟨u⟩
  |⟨upper⟩⟨v⟩
  |⟨upper⟩⟨w⟩
  |⟨upper⟩⟨x⟩
  |⟨upper⟩⟨y⟩
  |⟨upper⟩⟨z⟩
  ;
⟨digit⟩ = zero
  |one
  |two
  |three
  |four
  |five
  |six
  |seven
  |eight
  |nine
  ;
⟨num2-10⟩ = two
  |three
  |four
  |five
  |six
  |seven
  |eight
  |nine
  |ten
  ;
⟨punctuation⟩ = tab
  |back tab
  |enter
  |space
  |back space
  |period
  |dot
  |comma
  |question mark
  |exclamation point
  |at [sign]
  |dash|hyphen
  |slash
  |back slash
  |colon
  |semicolon
  |apostrophe
  |quote
  |pound
  |percent
  |ampersand
  |asterisk
  |open paren
  |close paren
  |greater than
  |less than
  |plus
  |minus
  |equals
  |delete
  ;
⟨word⟩ = (⟨lowerletter⟩|⟨upperletter⟩|⟨punctuation⟩|⟨digit⟩|
⟨capping⟩)*;
⟨move⟩ = move(right|left)⟨num2-10⟩(characters|words)
  |move(right|left)(one|a)(character|word)
  |move(up|down)⟨num2-10⟩lines
  |move(up|down)(one|a)line
  |(next|previous)field
  ;
public⟨spell⟩ = ⟨word⟩
  |⟨move⟩
  ;
```

Appendix B: The final voice-spelling grammar

```
#JSGF V1.0;
grammar voicespell4;
//Copyright (c) 2002 IBM Corp. All Rights Reserved.
<a> = alpha;
<b> = bravo;
<c> = charlie|cat;
<d> = delta|dog;
<e> = echo;
<f> = foxtrot|frank;
<g> = golf;
<h> = hotel;
<i> = india;
<j> = jane;
<k> = kilo;
<l> = lima;
<m> = mike;
<n> = november|north;
<o> = oscar;
<p> = paul|papa;
<q> = quebec|queen;
<r> = romeo;
<s> = sierra;
<t> = tango;
<u> = uniform|under;
<v> = victor;
<w> = water|whiskey;
<x> = x ray;
<y> = yankee|yoyo;
<z> = zulu;
<upper> = capital|uppercase;
<capslock> = caps lock;
<lower> = lowercase;
<capping> = <upper>
  |<lower>
  |<capslock>
  ;
<lowerletter> = [<lower>]<a>
  |[<lower>]<b>
  |[<lower>]<c>
  |[<lower>]<d>
  |[<lower>]<e>
  |[<lower>]<f>
  |[<lower>]<g>
  |[<lower>]<h>
  |[<lower>]<i>
  |[<lower>]<j>
  |[<lower>]<k>
  |[<lower>]<l>
  |[<lower>]<m>
  |[<lower>]<n>
  |[<lower>]<o>
  |[<lower>]<p>
  |[<lower>]<q>
  |[<lower>]<r>
  |[<lower>]<s>
  |[<lower>]<t>
  |[<lower>]<u>
  |[<lower>]<v>
  |[<lower>]<w>
  |[<lower>]<x>
  |[<lower>]<y>
  |[<lower>]<z>
  ;
<upperletter> = <upper><a>
  |<upper><b>
  |<upper><c>
  |<upper><d>
  |<upper><e>
  |<upper><f>
  |<upper><g>
  |<upper><h>
  |<upper><i>
  |<upper><j>
  |<upper><k>
  |<upper><l>
  |<upper><m>
  |<upper><n>
  |<upper><o>
  |<upper><p>
  |<upper><q>
  |<upper><r>
  |<upper><s>
  |<upper><t>
  |<upper><u>
  |<upper><v>
  |<upper><w>
  |<upper><x>
  |<upper><y>
  |<upper><z>
  ;
<digit> = zero
  |one
  |two
  |three
  |four
  |five
  |six
  |seven
  |eight
  |nine
  ;
<num2-10> = two
  |three
  |four
  |five
  |six
  |seven
  |eight
  |nine
  |ten
  ;
<punctuation> = tab
  |back tab
  |enter
  |space
  |back space
  |period
  |dot
  |<specialdot>
  |comma
  |question mark
  |exclamation point
  |at [sign]
  |dash|hyphen
  |slash
  |back slash
  |colon
```

```
   |semicolon
   |apostrophe
   |quote
   |pound
   |percent
   |ampersand
   |asterisk
   |open paren
   |close paren
   |greater than
   |less than
   |plus
   |minus
   |equals
   |delete
   ;
⟨specialdot⟩ = dot
   |(water dot)
   |(dot(com|edu|net|org|cat|charlie|oscar|echo|north))
   ;
⟨word⟩ = (⟨lowerletter⟩|⟨upperletter⟩|⟨punctuation⟩|⟨digit⟩|
⟨capping⟩)*;
⟨move⟩ = move(right|left)⟨num2-10⟩(characters|words)
   |move(right|left)(one|a)(character|word)
   |move(up|down)⟨num2-10⟩lines
   |move(up|down)(one|a)line
   |(next|previous)field
   ;
public⟨spell⟩ = ⟨word⟩
   |⟨move⟩
   ;
```

*Trademark or registered trademark of International Business Machines Corporation.

**Trademark or registered trademark of Palm Inc., Xerox Corporation, Communication Intelligence Corporation, or WebSurveyor Corporation.

Cited references

1. R. W. Marklin, G. G. Simoneau, and D. Hoffman, "Effects of Computer Keyboard Setup Parameters and User's Anthropometric Characteristics on Wrist Deviation and Typing Efficiency," *Proceedings of the Human Factors and Ergonomics Society 42nd Annual Meeting* (1998), pp. 876–880.
2. D. A. Norman and D. Fisher, "Why Alphabetic Keyboards Are Not Easy to Use: Keyboard Layout Doesn't Much Matter," *Human Factors* **24**, 509–519 (1982).
3. R. W. Soukoreff and I. S. MacKenzie, "Theoretical Upper and Lower Bounds on Typing Speed Using a Stylus and Soft Keyboard," *Behaviour and Information Technology* **14**, 370–379 (1995).
4. I. S. MacKenzie and L. Chang, "A Performance Comparison of Two Handwriting Recognizers," *Interacting with Computers* **11**, 283–297 (1999).
5. I. S. MacKenzie and S. X. Zhang, "The Immediate Usability of Graffiti," *Proceedings of Graphics Interface '97*, Canadian Information Processing Society, Toronto (1997), pp. 129–137.
6. I. S. MacKenzie, R. B. Nonnecke, J. C. McQueen, S. Riddersma, and M. Meltz, "A Comparison of Three Methods of Character Entry on Pen-Based Computers," *Proceedings of the Human Factors and Ergonomics Society 38th Annual Meeting* (1994), pp. 330–334.
7. A. Sears and R. Arora, "An Evaluation of Gesture Recognition for PDAs," in *Usability Evaluation and Interface Design: Cognitive Engineering, Intelligent Agents and Virtual Reality*, M. J. Smith, G. Salvendy, D. Harris, and R. J. Koubek, Editors, Lawrence Erlbaum Associates, Mahwah, NJ (2001), pp. 1–5.
8. M. D. Fleetwood, M. D. Byrne, P. Centgraf, K. Q. Dudziak, B. Lin, and M. Dmitryi, "An Evaluation of Text-Entry in Palm OS—Graffiti and the Virtual Keyboard," *Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting* (2002), pp. 617–621.
9. N. A. Kleid and M. A. Bonto, *Handwriting Recognition and Soft Keyboard Study*, Technical Report 29.3008, IBM Corporation, Raleigh, NC (1995).
10. Y. Zha and A. Sears, "Data Entry for Mobile Devices Using Soft Keyboards: Understanding the Effect of Keyboard Size," in *Usability Evaluation and Interface Design: Cognitive Engineering, Intelligent Agents and Virtual Reality*, M. J. Smith, G. Salvendy, D. Harris, and R. J. Koubek, Editors, Lawrence Erlbaum Associates, Mahwah, NJ (2001), pp. 16–20.
11. I. S. MacKenzie, S. X. Zhang, and R. W. Soukoreff, "Text Entry Using Soft Keyboards," *Behaviour and Information Technology* **18**, 235–244 (1999).
12. I. S. MacKenzie and S. X. Zhang, "The Design and Evaluation of a High-Performance Soft Keyboard," *Proceedings of the ACM Conference on Human Factors in Computing Systems–CHI '99* (1999), pp. 25–31.
13. J. R. Lewis, "Effect of Error Correction Strategy on Speech Dictation Throughput," *Proceedings of the Human Factors and Ergonomics Society 43rd Annual Meeting* (1999), pp. 457–461.
14. S. K. Card, T. P. Moran, and A. Newell, *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum, Hillsdale, NJ (1983).
15. For the full report, see J. R. Lewis, *Frequency Distributions for Names and Unconstrained Words Associated with the Letters of the English Alphabet*, Technical Report 29.3437, IBM Corporation, West Palm Beach, FL (2001).
16. M. W. Hartley and J. R. Lewis, *Conditional Probabilities for IBM Voice Browser Recognition of Letters of the Alphabet*, Technical Report 29.3421, IBM Corporation, West Palm Beach, FL (2001).
17. For the full report, see J. R. Lewis and P. M. Commarford, *Developing and Tuning a Voice Spelling Alphabet for Devices with Small Displays*, Technical Report 29.3517, IBM Corporation, Boca Raton, FL (2002).

Accepted for publication May 30, 2003.

James R. Lewis *IBM Pervasive Computing Division, 8051 Congress Avenue, Suite 2227, Boca Raton, Florida 33487 (jimlewis@us.ibm.com).* Dr. Lewis received his M.A. degree in engineering psychology from New Mexico State University in 1982 and his Ph.D. in psycholinguistics from Florida Atlantic University in 1996. He has worked for IBM as a human factors engineer since 1981. Currently a senior human factors engineer in IBM Voice Systems, his recent product-related work and research interest has been in the design of interactive voice response systems that use speech recognition technologies. Dr. Lewis is a member of the Human Factors and Ergonomics Society and the American Psychological Society, is certified as a Human Factors Professional by the Board of Certification in Ergonomics, and serves on the editorial board of the *International Journal of Human-Computer Interaction*.

Patrick M. Commarford *IBM Pervasive Computing Division, 8051 Congress Avenue, Suite 2228, Boca Raton, Florida 33487* *(commarfo@us.ibm.com).* Mr. Commarford is pursuing his Ph.D. in applied experimental and human factors psychology at the University of Central Florida. He is currently working as a human factors engineer for IBM Voice Systems. His professional interests include usability, human-computer interaction, and user interface design.